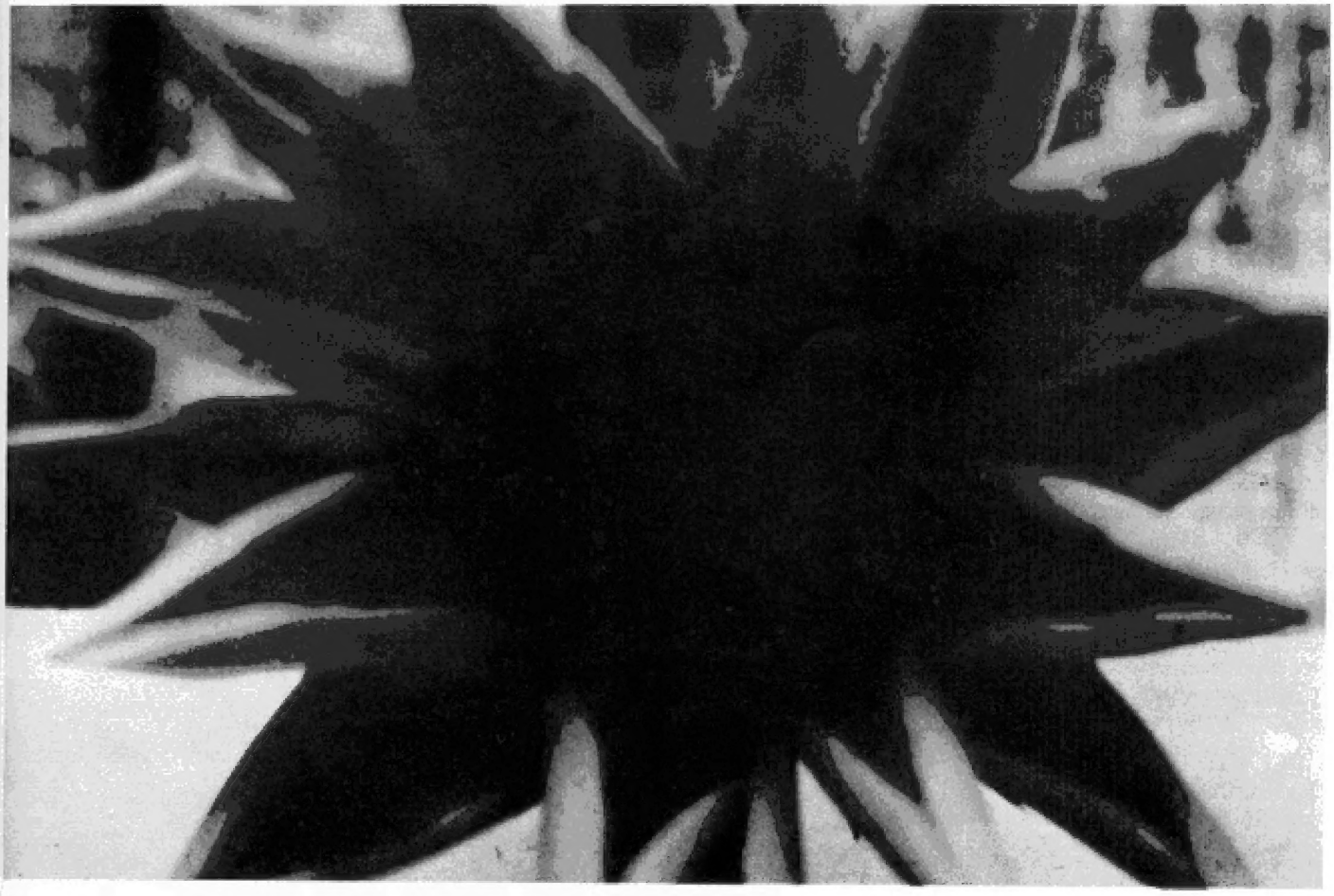
INTERFACE

Dicas de hardware

Carlos Teixeira Dias.

TCI - a interface que testa circuitos integrados

O TCI é uma interface para microcomputadores baseados no microprocessador Z-80, que permite testar qualquer tipo de circuito integrado através de programas específicos, desenvolvidos pelo próprio usuário. Foi desenvolvida por Carlos Teixeira Dias, estudante de engenharia do Instituto Nacional de Telecomunicações (Inatel), de Santa Rita de Sapucaí, com o auxílio de seus professores. O artigo foi dividido em duas partes: a primeira descreve o software utilizado para implementar o interface; a segunda descreve o hardware da interface.



DESCRIÇÃO DO SOFTWARE

O TCI (Testador de Circuito Integrado) necessita de uma rotina em Basic, a qual terá a função de identificar o tipo de CI a ser testado e executar a sua própria sub-rotina que se encontra em linguagem de máquina.

No entanto, para escrevê-la necessitamos de uma área específica de memória e, para reservá-la podemos utilizar dois métodos.

O primeiro é indicar ao microcomputador o endereço que será o topo da memória, ficando então liberada uma área acima deste endereço na qual poderemos escrever as sub-rotinas em linguagem de máquina.

Supondo que queremos reservar uma área acima do endereço 30000 (de 30000 a 32767), para isto deveremos transformar 30000 em binário e dividi-lo em dois bytes; o primeiro será o byte de baixa ordem (L) e o segundo o byte de alta ordem (H). Então teremos: L = 48 e H = 117

Estes dois bytes deverão ser escritos nos endereços 16388 e 16389, que são os endereços que especificam ao microcomputador onde está localizado o topo da memória (RAMTOP). Logo, teremos:

```
POKE 16388,48
POKE 16389,117
NEW
```

Após darmos o NEW, teremos do endereço 30000 ao 32767 livres para escreyer qualquer programa em linguagem de máquina. A única desvantagem de utilizar este método é que as sub-rotinas que se encontram acima do endereço 30000 não poderão ser gravadas em fitas cassetes.

O segundo método consiste em utilizarmos a instrução REM, precedida de uma série de caracteres alfanuméricos. O número de caracteres deverá ser igual ao número de bytes utilizados pelo programa em linguagem de máquina e, então, teremos os endereços livres a partir do endereço 16514, como no exemplo:

REM 1 2 3 4 5 S D CASA HOJE. . . . . etc.

A rotina em BASIC a ser utilizada é mostrada a seguir, ficando a cargo do leitor efetuar algumas mudanças na estrutura da mesma, caso seja necessário para a finalidade que o mesmo possa vir a utilizar.

```
1 REM 1 2 3 4 . . . etc
10 PRINT "ENTRAR COM O NÚMERO DO CIRCUITO INTEGRADO A SER TESTADO"
20 INPUT A$
30 IF A$ = "7404" THEN GOTO 400
40 IF A$ = "7405" THEN GOTO 400
50 IF A$ = "7408" THEN GOTO XXX
60 IF A$ = "7410" THEN GOTO YYY
. . . . . .
. . . . . .
100 PRINT "ESTE CIRCUITO INTEGRADO NÃO PODE SER TESTADO, DEVIDO A FALTA DA SUB-ROTINA DE TESTE DO MESMO"
110 GOTO 10
120 LET A = PEEK 32756
130 LET B = PEEK 32757
140 LET C = PEEK 32758
150 LET D = PEEK 32759
160 IF A = 1 THEN GOTO 180
170 GOTO 230
180 IF B = 1 THEN GOTO 200
190 GOTO 230
200 IF C = 1 THEN GOTO 220
210 GOTO 230
220 IF D = 1 THEN GOTO 240
230 PRINT "CIRCUITO INTEGRADO SE ENCONTRA DANIFICADO"
240 PRINT "CIRCUITO INTEGRADO EM PERFEITAS CONDIÇÕES"
250 RETURN
260 . . . . . .
270 . . . . . .
. . . . . .
. . . . . .
```

No programa acima, as linhas 30 e 40 perguntam se A$ é igual a 7404 ou 7405. Caso seja, o computador passa a executar imediatamente a linha 400, onde se encontra a sub-rotina de teste destes circuitos integrados.

Os X's e os Y's que se encontram no final das linhas 50 e 60 indicam, respectivamente, o número das linhas em que deverá iniciar a sub-rotina de teste dos CI's 7408 e 7410, respectivamente.

Os endereços 32756, 32757, 32758 e 32759 são utilizados pela sub-rotina em linguagem de máquina para armazenar resultados obtidos durante a comparação (teste) do CI 7404 ou 7405. Dependendo do tipo de CI a ser testado, serão utilizados mais endereços para armazenar estes resultados obtidos durante os testes. Normalmente utiliza-se dois endereços para cada tipo de combinação a ser testada, por exemplo: para o 7404 e 7405 temos duas combinações possíveis.

Entrada	Saída
0	1
1	0

Logo, necessitaremos de 4 endereços para armazenar os dados.

Para o 7408 temos quatro tipos de combinações:

E1	E2	S
0	0	0
0	1	0
1	0	0
1	1	1

Logo, necessitaremos de 8 endereços para armazenar os dados. Para que não tenhamos que modificar a rotina em Basic faremos uma outra sub-rotina em Basic para efetuar a análise dos endereços utilizados durante o teste de cada tipo de CI.

A partir da linha 260 deve-se colocar as outras sub-rotinas (em Basic), que verificarão os endereços utilizados durante o teste para que possamos obter na tela do vídeo um resultado real do teste realizado.

Façamos agora a sub-rotina em linguagem de máquina, a qual é responsável pelo teste do circuito integrado; logo após o programa faremos uma análise detalhada da mesma.

```
400 LET R = 0
405 POKE 16514,33
410 POKE 16515,212
415 POKE 16516,255
420 POKE 16517,54
425 POKE 16518,85
430 POKE 16519,33
435 POKE 16520,2
440 POKE 16521,254
445 POKE 16522,54
450 POKE 16523,207
455 POKE 16524,54
460 POKE 16525,85
465 POKE 16526,6
470 POKE 16527,168
475 POKE 16528,33
480 POKE 16529,0
485 POKE 16530,254
490 POKE 16531,54
495 POKE 16532,128
500 POKE 16533,33
505 POKE 16534,3
510 POKE 16535,254
515 POKE 16536,54
520 POKE 16537,207
525 POKE 16538,54
530 POKE 16539,43
535 POKE 16540,14
540 POKE 16541,212
545 POKE 16542,33
550 POKE 16543,1
555 POKE 16544,254
560 POKE 16545,54
565 POKE 16546,212
570 POKE 16547,33
575 POKE 16548,0
```

```
580 POKE 16549,254
585 POKE 16550,126
590 POKE 16551,184
595 POKE 16552,54
600 POKE 16553,244
605 POKE 16554,127
610 POKE 16555,194
615 POKE 16556,178
620 POKE 16557,64
625 POKE 16558,54
630 POKE 16559,1
635 POKE 16560,195
640 POKE 16561,180
645 POKE 16562,64
650 POKE 16563,54
655 POKE 16564,0
660 POKE 16565,33
665 POKE 16566,1
670 POKE 16567,254
675 POKE 16568,126
680 POKE 16569,185
685 POKE 16570,33
690 POKE 16571,245
695 POKE 16572,127
700 POKE 16573,194
705 POKE 16574,196
710 POKE 16575,64
715 POKE 16576,54
720 POKE 16577,1
725 POKE 16578,195
730 POKE 16579,198
735 POKE 16580,64
740 POKE 16581,54
745 POKE 16582,0
750 POKE 16583,22
755 POKE 16584,84
760 POKE 16585,33
765 POKE 16586,0
770 POKE 16587,254
775 POKE 16588,54
780 POKE 16589,0
785 POKE 16590,30
790 POKE 16591,170
795 POKE 16592,33
800 POKE 16593,1
805 POKE 16594,254
810 POKE 16595,54
815 POKE 16596,128
820 POKE 16597,33
825 POKE 16598,0
830 POKE 16599,254
835 POKE 16600,126
840 POKE 16601,186
845 POKE 16602,33
850 POKE 16603,246
855 POKE 16604,127
860 POKE 16605,194
865 POKE 16606,228
870 POKE 16607,64
875 POKE 16608,54
880 POKE 16609,1
885 POKE 16610,195
890 POKE 16611,230
895 POKE 16612,64
900 POKE 16613,54
905 POKE 16614,0
910 POKE 16615,33
915 POKE 16616,1
920 POKE 16617,254
925 POKE 16618,126
930 POKE 16619,187
935 POKE 16620,33
940 POKE 16621,247
945 POKE 16622,127
950 POKE 16623,194
955 POKE 16624,245
960 POKE 16625,64
965 POKE 16626,54
970 POKE 16627,1
975 POKE 16628,195
980 POKE 16629,247
985 POKE 16630,64
990 POKE 16631,54
995 POKE 16632,0
1000 POKE 16633,33
1005 POKE 16634,0
1010 POKE 16635,255
1015 POKE 16636,54
1020 POKE 16637,0
1025 POKE 16638,201
1030 RAND USER 16514
1035 LET R = R + 1
1040 IF R = 5 THEN GOTO 1050
1045 GOTO 405
1050 GOSUB 120
1055 GOTO 10
```

ANÁLISE DO PROGRAMA EM LINGUAGEM DE MÁQUINA

Da linha 405 a 425 o microcomputador habilita o decodificador 1, figura 1, através dos bits de alta ordem (A8 a A15); carrega o primeiro conjunto de flip-flops através dos bits de baixa ordem (A0 a A7) e carrega o segundo conjunto de flip-flops através do barramento de dados (D0 a D7). Os dados armazenados nestes conjuntos de flip-flops são responsáveis pela indicação de entrada ou saída de dados nos amplificadores de corrente.

Da linha 430 a 450 o microcomputador habilita o decodificador 2, figura 1, através do byte H (A8 a A15); o byte L (A0 a A7) habilita o controle da porta A na PIO-Z80 e, através do barramento de dados habilita o controle bit a bit na porta A da PIO-Z80.

Da linha 455 a 460 o microcomputador habilita os *tri-states* internos da porta A da PIO-Z80, especificando se cada saída da porta trabalhará como entrada ou como saída de dados ("1" (um) saída de dados e "0" (zero) entrada de dados).

Da linha 465 a 470 carrega o registro B do microcomputador com os dados que deverão ser obtidos do CI a ser testado.

Da linha 475 a 495 habilita o decodificador 2 através do byte H e, através do byte L habilita o registro A da porta A da PIO-Z80. Através do barramento de dados armazena no registro A a combinação a ser testada no CI (terminal 1 ao 8).

Da linha 500 a 520 o decodificador 2 é habilitado através de H; L habilita o controle da porta B da PIO-Z80 e, através do barramento de dados habilita o controle de bit a bit na porta B.

Da linha 525 a 530 os *tri-states* internos da porta B da PIO-Z80 são habilitados.

Da linha 535 a 540 é armazenado no registro C do microcomputador o dado correto a ser obtido do CI durante o teste.

Da linha 545 a 565 o decodificador 2 é habilitado através de H; o registro da porta B é habilitado através do byte L e através do barramento de dados armazena na porta B da PIO-Z80 a combinação a ser testada no CI (do terminal 9 ao 16).

Da linha 570 a 585, H habilita o decodificador 2; L habilita o registro da porta A da PIO-Z80 e realiza a leitura do registro A da PIO, armazenando o conteúdo da mesma no acumulador do microcomputador.

Da linha 590 a 605 é realizada uma comparação do dado contido no acumulador com o dado contido no registro B do microcomputador ("0" (zero) para igualdade e "1" (um) para não-igualdade), carregando o par de registradores HL com o endereço 32756.

Da linha 610 a 655, se o valor do acumulador após a comparação for "0" (zero) coloca "1" (um) no endereço 32756.

Da linha 660 a 675, H habilita o decodificador 2; L habilita o registro da porta B e realiza a leitura do registro da porta B.

Da linha 680 a 695 compara o valor contido no acumulador com o registro C do microcomputador e carrega os registradores HL com o endereço 32757.

Da linha 700 a 745, caso o valor do acumulador seja "0" (zero) coloca "1" (um) no endereço 32757 e, sendo "1" (um) coloca "0" (zero) no endereço 32757.

Da linha 750 a 755 o registro D do microcomputador é carregado com o

dado correto a ser obtido do CI durante o teste.

Da linha 760 a 780, H habilita o decodificador 2; L habilita o registro da porta A e através do barramento de dados escreve no registro da porta A da PIO-Z80 uma nova combinação a ser testada.

Da linha 785 a 790 carrega o registro E do microcomputador com o dado correto a ser obtido durante o teste do CI.

Da linha 795 a 815, H habilita o decodificador 2; L habilita o registro da porta B e através do barramento de dados armazena um novo dado para efetuar o teste do CI.

Da linha 820 a 835, H habilita o decodificador 2; L habilita o registro da porta A da PIO e realiza a leitura do registro da porta A da PIO-Z80.

Da linha 840 a 855 compara o acumulador com o registro do microcomputador e carrega o par de registradores HL com o endereço 32758.

Da linha 860 a 905, caso o valor contido no acumulador após a comparação seja "0" (zero), coloca "1" (um) no endereço 32758 e se for "1" (um) coloca "0" (zero) no endereço 32758.

Da linha 910 a 925, H habilita o decodificador 2; L habilita o registro da porta B e realiza a leitura da porta B da PIO-Z80.

Da linha 930 a 945 compara o valor do acumulador com o registro E do microcomputador e carrega o par de registradores HL com o endereço 32759.

Da linha 950 a 995, caso o valor do acumulador após a comparação seja "0" (zero) coloca "1" (um) no endereço 32759 e, caso seja "1" (um) coloca "0" (zero) no endereço 32759.

Da linha 1000 a 1020 reseta os dois conjuntos de flip-flops.

Na linha 1025 informa ao microcomputador para retornar ao Basic.

Na linha 1030 indica ao microcomputador que a partir do endereço 16514 ao endereço anterior (16638) deve ser interpretado como um programa em linguagem de máquina.

As linhas 400, 1035, 1040 e 1045 são responsáveis por um ciclo de 5 testes seguidos no CI, antes de apresentar o resultado final no vídeo.

Caso o leitor esteja familiarizado com o ASSEMBLER (*linguagem de máquina*) o mesmo poderá modificar a estrutura da sub-rotina de teste, de maneira que o final do teste apresente um resultado mais detalhado sobre as condições do CI; caso o mesmo se encontre danificado, poderá ser indicado qual ou quais as partes do mesmo que se encontram defeituosas. O programa em ASSEMBLER é apresentado a seguir.

PROGRAMA EM ASSEMBLER
SUB-ROTINA DE TESTE DO 7404 e 7405

Endereço	Código	Instrução	Byte H	Byte L	Comentário
16514	33	LD HL, NN	11111111	11010100	H - Habilita decodificador 1 L - Carrega Flip-Flop 1
16515	212				
16516	255				
16517	54	LD (HL), N		01010101	Carrega Flip-Flop 2
16518	85				
16519	33	LD HL, NN	11111110	00000010	H - Habilita decodificador 2 L - Habilita controle A
16520	2				
16521	254				
16522	54	LD (HL), N		11001111	Habilita em A controle bit a bit
16523	207				
16524	54	LD (HL), N		01010101	Habilita tri-state interno de A
16525	85				
16526	6	LD B, N		10101000	Carrega registro B com dado correto a ser comparado
16527	168				
16528	33	LD HL, NN	11111110	00000000	H - Habilita decodificador 2 L - Habilita registro A
16529	0				
16530	254				
16531	54	LD (HL), N		10101000	Escreve no registro A
16532	168				
16533	33	LD HL, NN	11111110	00000011	H - Habilita decodificador 2 L - Habilita controle B
16534	3				
16535	254				
16536	54	LD (HL), N		11001111	Habilita em B controle bit a bit
16537	207				
16538	54	LD (HL), N		00101011	Habilita tri-state interno de B
16539	43				
16540	14	LD C, N		11010100	Carrega registro C com dado correto a ser comparado
16541	212				
16542	33	LD HL, NN	11111110	00000001	H - Habilita decodificador 2 L - Habilita registro B
16543	1				
16544	254				
16545	54	LD (HL), N		11010100	Escreve no registro B
16546	212				
16547	33	LD HL, NN	11111110	00000000	H - Habilita decodificador 2 L - Habilita registro 2
16548	0				
16549	254				
16550	126	LD A, (HL)			Lê conteúdo do Registro A (PIO-Z80)

```
16551  184  CP B                                          Compara Registro A com Registro B (uP)
16552  54   LD HL, NN                                     Carrega HL com endereço 32756
16553  244
16554  127
16555  194  JPNZ NN                                      Se não for zero GOTO 16562
16556  178                                                Se for zero prossegue
16557  64
16558  54   LD (HL), N    00000001                        Coloca 1 no endereço 32756
16559  1
16560  195  JP NN                                         GOTO endereço 16564
16561  180
16562  64
16563  54   LD (HL), N    00000000                        Coloca 0 no endereço 32756
16564  0
16565  33   LD HL, NN     11111110  00000001              H - Habilita decodificador 2
                                                          L - Habilita registro B
16566  1
16567  254
16568  126  LD A, (HL)                                   Lê conteúdo do Registro B (PIO)
16569  185  CP C                                          Compara Registro B com Registro C (uC)
16570  33   LD HL, NN                                     Carrega HL com endereço 32757
16571  245
16572  127
16573  194  JPNZ NN                                      Se não for zero GOTO 16580
16574  196                                                Se for zero prossegue
16575  64
16576  54   LD (HL), N    00000001                        Coloca 1 no endereço 32757
16577  1
16578  195  JP NN                                         GOTO 16582
16579  198
16580  64
16581  54   LD (HL), N    00000000                        Coloca 0 no endereço 32757
16582  0
16583  22   LD D, N       01010100                        Carrega Registro D com dado correto a ser comparado
16584  84
16585  33   LD HL, NN     11111110  00000000              H - Habilita decodificador 2
                                                          L - Habilita registro A (PIO)
16586  0
16587  254
16588  54   LD (HL), N    00000000                        Escreve no Registro A da PIO-Z80
16589  0
16590  30   LD E, N       10101010                        Carrega Registro E com dado correto a ser comparado
16591  170
16592  33   LD HL, NN     11111110  00000001              H - Habilita decodificador 2
                                                          L - Habilita Registro B da PIO-Z80
16593  1
16594  254
16595  54   LD (HL), N    10000000                        Escreve no Registro B da PIO-Z80
16596  128
16597  33   LD HL, NN     11111110  00000000              H - Habilita decodificador 2
                                                          L - Habilita Registro A
16598  0
16599  254
16600  126  LD A, (HL)                                   Lê conteúdo do Registro A da PIO
16601  186  CP D                                         Compara Registro D com Registro A da PIO
16602  33   LD HL, NN                                     Carrega HL com endereço 32758
16603  246
16604  127
16605  194  JPNZ NN                                      Se não for zero GOTO 16612
16606  228                                                Se for zero prossegue
16607  64
16608  54   LD (HL), N    00000001                        Coloca 1 no endereço dado por HL
16609  1
16610  195  JP NN                                         GOTO 16614
16611  230
16612  64
16613  54   LD (HL), N    00000000                        Coloca 0 no endereço 32758
16614  0
16615  33   LD HL, NN     11111110  00000001              H - Habilita decodificador 2
                                                          L - Habilita Registro B
16616  1
16617  254
16618  126  LD A, (HL)                                   Lê conteúdo do Registro B da PIO
16619  187  CP E                                          Compara Registro E com Registro B da PIO
16620  33   LD HL, NN                                     Carrega HL com endereço 32759
16621  247
16622  127
16623  194  JPNZ NN                                      Se não for zero GOTO 16629
16624  245                                                Se for zero prossegue
16625  64
16626  54   LD (HL), N    00000001                        Coloca 1 no endereço 32759
16627  1
16628  195  JP NN                                         GOTO 16631
16629  247
16630  64
16631  54   LD (HL), N    00000000                        Coloca zero no endereço 32759
16632  0
16633  33   LD HL, NN     11111111  00000000              H - Habilita decodificador 1
                                                          L - Carrega Flip-Flop 1
16634  0
16635  255
16636  54   LD (HL), N    00000000                        Carrega Flip-Flop 2 (Reset nos Flip-Flops)
16637  0
16638  201  RET                                           Retorna para ROTINA BASIC
```

DESCRIÇÃO DE HARDWARE

O hardware utilizado no testador de CI's é bastante simples e tem como elemento principal a PIO-Z80, que é um importantíssimo dispositivo de entrada/saída. A figura 1 apresenta o diagrama em blocos do Testador de Circuito Integrado.

Vamos fazer uma pequena análise do funcionamento da PIO-Z80, para que o leitor possa compreender melhor o funcionamento do circuito e do programa em linguagem de máquina.

A PIO-Z80 é constituída por duas portas A e B, sendo que cada porta possui um registro de dados e controles. Os controles são responsáveis pela definição se as portas trabalharem como saídas ou entradas, ou ainda, se trabalharem com controle bit a bit. Neste caso, deve-se habilitar os *tri-states* internos das portas, especificando quais deles serão saída e quais serão entrada. O registro de dados tem a função de armazenar os dados provenientes do barramento de dados, após ser habilitado, de modo que possa ser obtido na saída da porta durante o tempo desejado.

As entradas B/$\overline{A}$ e C/$\overline{D}$ da PIO-Z80 são ligadas aos bits A0 e A1 do barramento de endereço do microcomputador e selecionam em que porta trabalharão, porta A ou porta B. Assim sendo, utilizamos o byte L (byte de baixa ordem) para habilitar e controlar as

BYTE L	FUNÇÃO
	Habilita registro da porta A
	Habilita registro da porta B
	Habilita controle de A
	Habilita controle de B

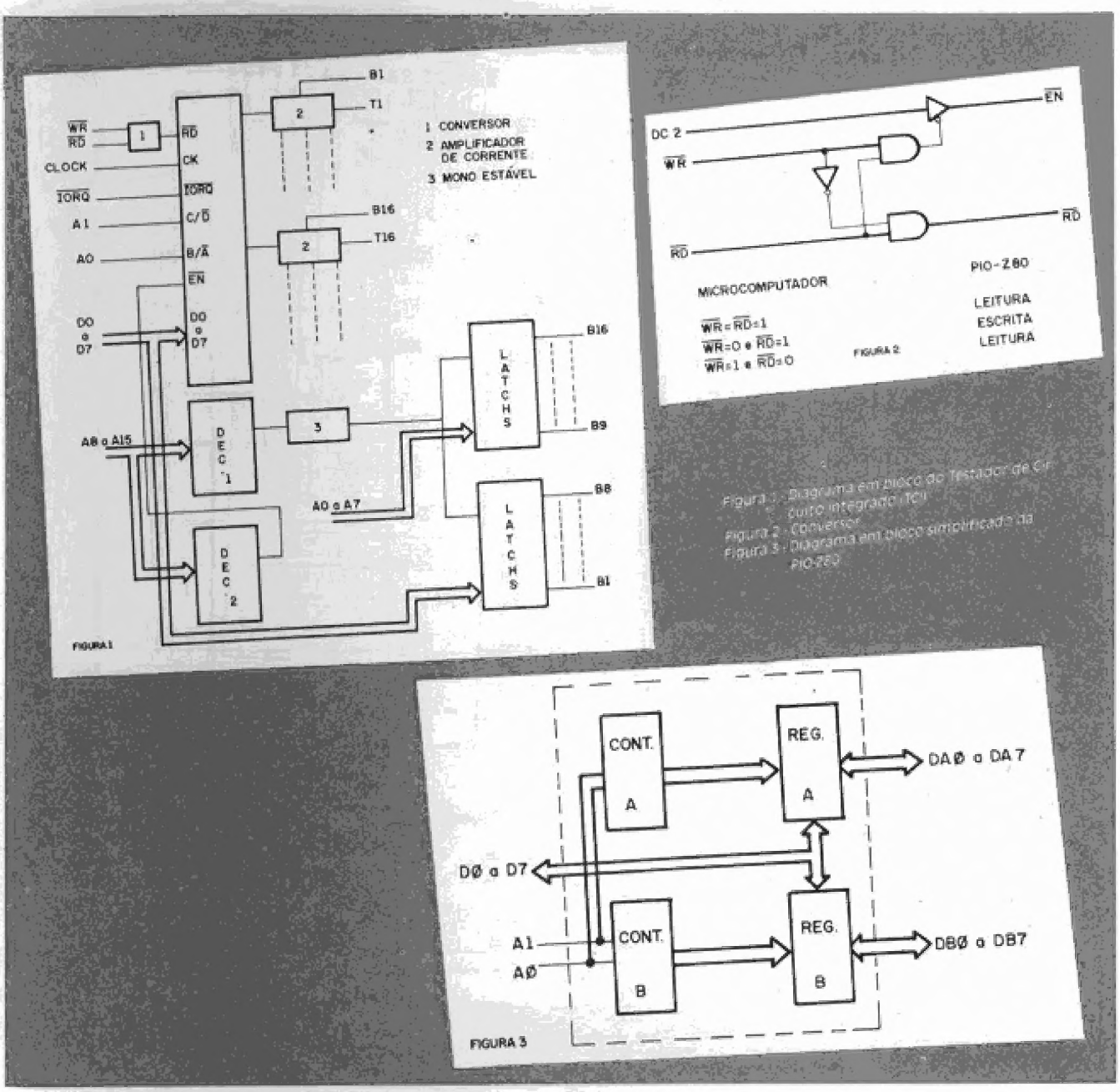


Figura 1 - Diagrama em bloco do Testador de Circuito Integrado (TCI)
Figura 2 - Conversor
Figura 3 - Diagrama em bloco simplificado da PIO-Z80

Para indicar uma operação de leitura ou escrita na PIO-Z80 necessita-se de um pequeno circuito conversor, pois o microcomputador utiliza duas saídas para indicar uma operação de leitura ou escrita (WR e RD) e a PIO-Z80 só tem uma entrada (RD) para indicar estas operações. A figura 2 apresenta o circuito conversor e a sua tabela verdade.

Note que no circuito da figura 2 o sinal EN é ligado à entrada de habilitação da PIO, que só é habilitada quando WR ou RD são diferentes de 1, isto é, a PIO só é habilitada durante uma operação de leitura ou escrita.

Para a gravação de dados na porta A da PIO-Z80 utiliza-se os seguintes passos:

1º Passo: Habilita-se o controle de A, através do byte L (byte de baixa ordem), com o endereço 00 00 00 10.

2º Passo: Habilita-se o controle bit a bit em A, através do barramento de dados, por 11 XX 11 11, onde XX representa qualquer estado lógico.

3º Passo: Habilita-se os *tri-states* internos da porta A, através do barramento de dados, "1" (um) para saída de dados e "0" (zero) para entrada de dados.

4º Passo: Habilita-se o registro A, através do byte L, com o endereco 00 00 00 00.

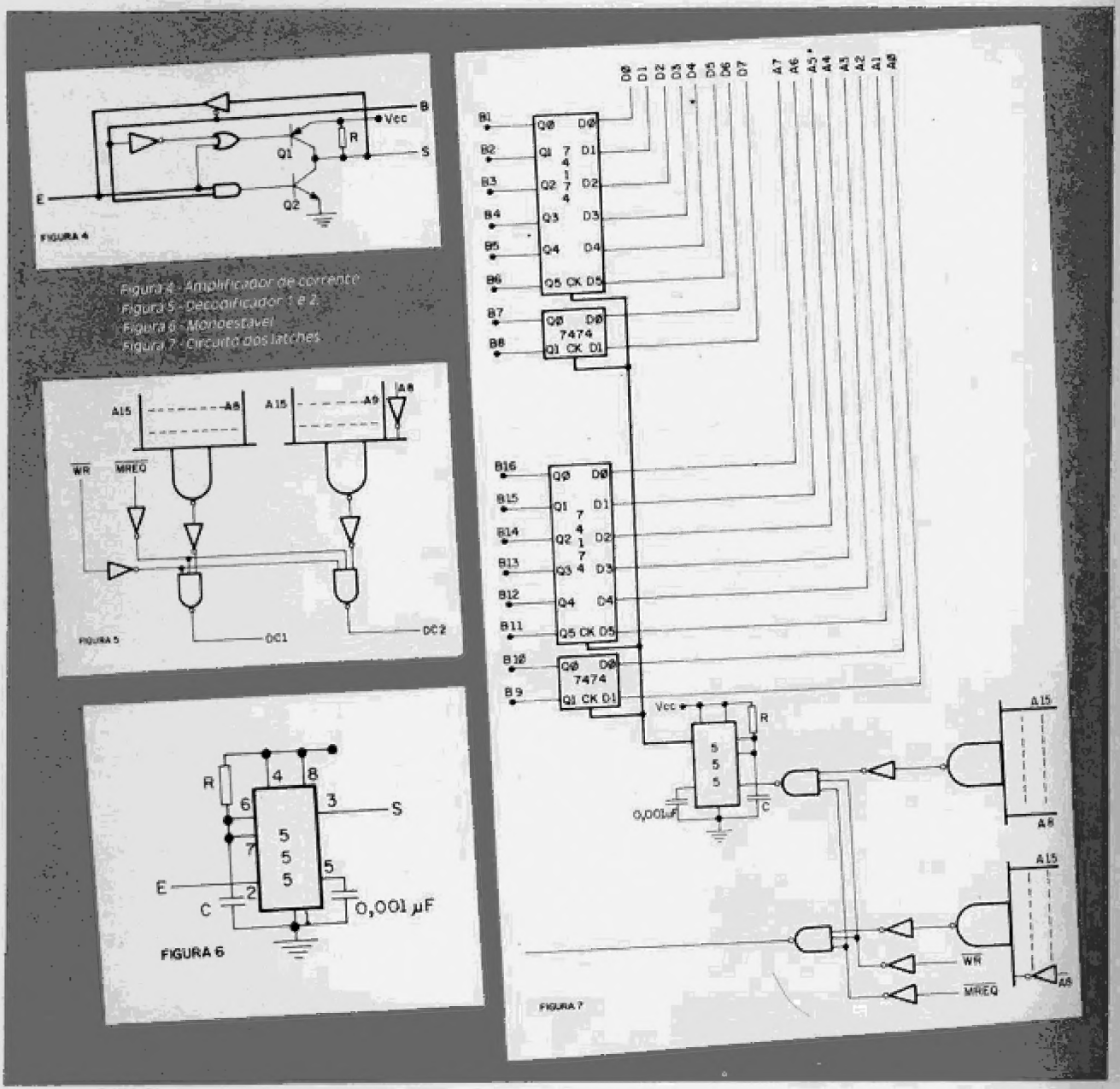
5º Passo: Escreve-se o dado a ser utilizado no registro A.

Para gravar dados na porta B deve-se mudar os passos 1 e 4.

A figura 3 apresenta o diagrama em bloco interno (simplificado) da PIO-Z80.

Outra parte importante a ser mencionada é o valor de corrente que a PIO pode fornecer. Como alguns CI's consomem até 120 mA e a PIO não consegue fornecer esta corrente, necessita-se de um amplificador de corrente. A figura 4 apresenta o circuito de um amplificador de corrente.

No circuito onde Q1 é um transistor PNP e Q2 é NPN, pode-se utilizar qualquer transistor que forneça uma corrente em torno de 150 mA. O resistor R é utilizado para testar CI's de coletor aberto, no caso do 7405. A entrada E é a entrada do amplificador de



corrente e provém das portas da PIO; S é a saída do amplificador de corrente, sendo conectada diretamente aos terminais do Circuito Integrado a ser testado; e B provém dos flip-flops(latches), sendo responsável pela definição do amplificador de corrente como uma entrada ou saída de dados.

Devido ao fato de que os terminais de alimentação dos CI's a serem testados variam de CI para CI, necessita-se repetir o circuito de amplificação de corrente para cada saída de cada porta da PIO-Z80 (16 vezes).

Os decodificadores 1 e 2 são constituídos por um conjunto de duas portas NAND de oito entradas, cinco portas inversoras e duas portas NAND de três entradas. A figura 5 apresenta os decodificadores 1 e 2.

A saída do decodificador 1 é conectada à entrada de um monoestável, de maneira que quando for realizada a decodificação o monoestável será ativado, produzindo um pulso de nível alto e de pequena duração. Este pulso é utilizado pelos flip-flops para carregar os dados, que se encontram em suas entradas e necessitam permanecer por um determinado tempo, após o pulso do monoestável ter retornado ao nível "0".

Pode-se utilizar como monoestável o 555. A figura 6 apresenta uma configuração; note que R e C são calculados em função do tempo em que o pulso deve permanecer em nível alto.

O circuito dos latchs é constituído de flip-flops tipo D, por serem de fácil aquisição e de baixo custo. Como são necessários dois conjuntos de flip-flops, onde cada conjunto utiliza oito flip-flops, optou-se por utilizar dois 74174 e dois 7474. A figura 7 apresenta o circuito dos latchs (também poderia utilizar-se dois 8212).

A figura 8 apresenta o diagrama elétrico de todo o circuito do Testador de Circuito Integrado (TCI). O leitor que possuir um conhecimento razoável de eletrônica digital poderá implementar e ampliar este Testador de Circuitos Integrados, de maneira que o mesmo possa testar CI's de 24 pinos. Para isso aconselhamos a utilização de uma outra PIO-Z80.

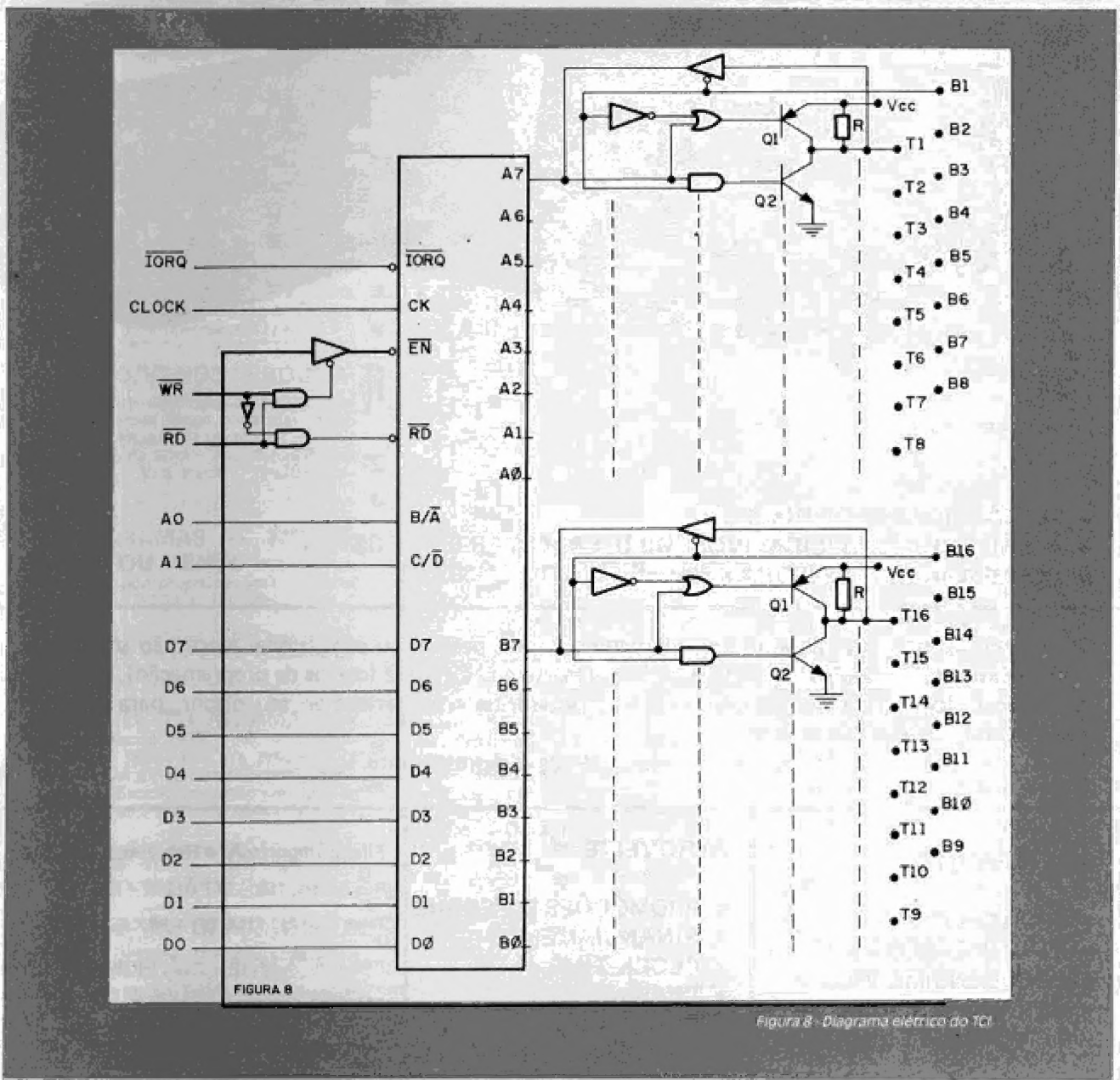


Figura 8 - Diagrama elétrico do TCI